

AULA 2

**HTML,
CSS
&
JAVASCRIPT**



A linguagem da web

HTML, CSS e JavaScript

Como funciona a HTML e as CSS?

Pode ser estranho para você ler “a HTML” e também “as CSS”. Muita gente trata as duas tecnologias com o artigo masculino na maioria das vezes. Na verdade, isso nem está totalmente errado, mas eu sempre vou considerar defini-las pela tradução dos seus termos.

Para que serve HTML, CSS e JS?

HTML	Conteúdo
CSS	Estilo
JAVASCRIPT	Interatividade

Guarde bem a tabela anterior sempre que você precisar decidir qual linguagem vai utilizar em cada situação.

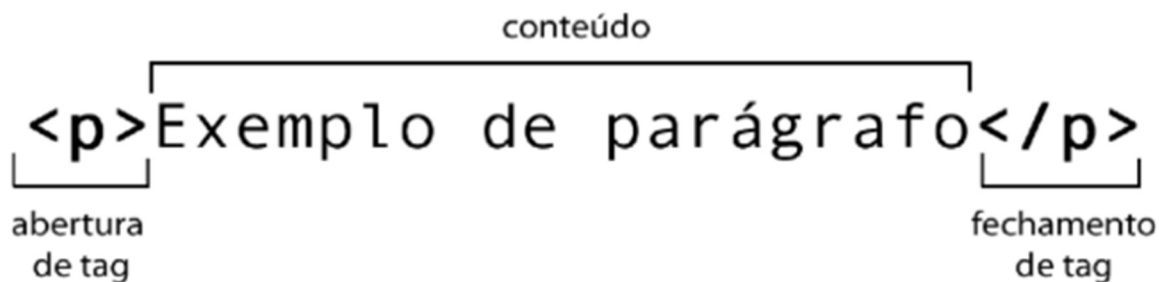
Ao abrir uma determinada notícia, você vai ver o texto, as imagens, os vídeos e todo aquele conteúdo que compõe a notícia em si. Isso tudo foi criado em **HTML**. Ela é focada em **conteúdo**.

Agora preste atenção nas cores, na posição dos componentes e organização visual do conteúdo em colunas, blocos visuais e tudo mais. Tudo foi definido em **CSS**. Ela é focada no **design/estilo**.

Finalmente, provavelmente existe o menu do site. Quando você clica nele, acontece uma animação. Ao mover o mouse sobre as sessões, é possível que aconteçam algumas interações interessantes. Isso foi desenvolvido com ajuda de **Java Script**. Ela é uma linguagem focada nas **interações**.

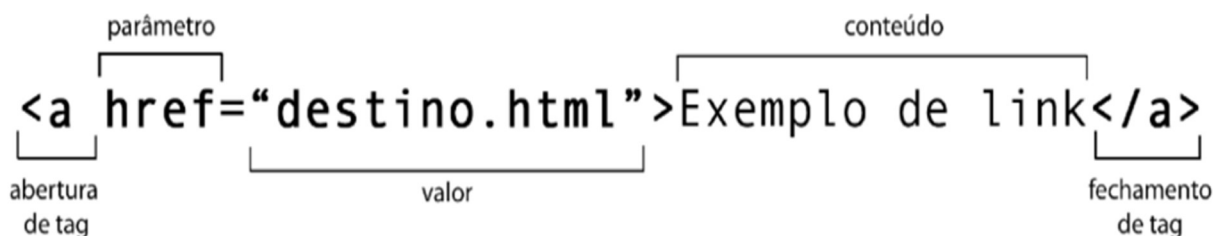
Tags HTML, aí vamos nós...

Como podemos ver, a HTML funciona baseada em marcações específicas chamadas **tags**. Uma tag é um conjunto de palavras entre sinais de **colchete angular**, conforme representado a seguir.



Na imagem anterior, você consegue perceber o uso da tag `<p>` para a criação de um parágrafo simples. A maioria das tags possuem uma **abertura** e um **fechamento**, e você identifica isso pela presença da barra no fechamento da tag.

Além disso, as tags também podem ter atributos e valores, que vão configurar seu comportamento:



Uma mesma tag pode ter vários parâmetros, cada um com seu valor. Entretanto, algumas tags não possuem a necessidade de **conteúdo** interno e por isso não possuem fechamento. É o caso, por exemplo, das tags `
` e ``. Isso é algo natural, não se preocupe com isso agora.

Chegou a vez dos seletores CSS

As **CSS** são **Cascading Style Sheets** (Folhas de Estilo em Cascata). Elas são usadas para configurar um **resultado visual** dos elementos HTML.

As configurações das CSS são realizadas através dos **seletores**. Vamos ver a anatomia de um seletor.

```
seletor
┌ p {
└   ┌─────────── declaração ───────────┐
    font-family: Arial;
    font-size: 12pt;
    ┌ color: ──┐ ┌─── blue; ──┐
    └───┬───┐ └───┬───┐
        └─┬──┘ └──┬──┘
          propriedade valor
    └──────────┘
}
```

O seletor apresentado anteriormente vai configurar o visual dos elementos de parágrafo do site corrente. O uso das chaves delimita todas as declarações relativas ao seletor atual. No seletor que eu te mostrei, serão feitas três configurações:

- A fonte escolhida foi Arial.
- O tamanho da letra será 12pt (pontos).
- A cor da letra será azul.

Note que, ao final de cada **declaração**, temos que colocar ponto-e-vírgula para indicar que ela se encerrou.

Estrutura básica de um documento HTML

Note Ao criar um novo documento HTML, devemos sempre escrever a estrutura básica de um documento desse formato. Vamos analisar cada uma das 11 linhas que compõem esse documento base.

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width,
6          initial-scale=1.0">
7      <title>Document</title>
8  </head>
9  <body>
10
11 </body>
    </html>
```

- **Linha 1:** Indica que o documento atual será escrito na versão mais atualizada da linguagem (no caso, HTML5)
- **Linhas 2 e 11:** Delimitam o documento HTML, que é sempre dividido em duas partes: a cabeça e o corpo. Na linha 2, também estamos indicando que o conteúdo desse site será no idioma Português do Brasil.
- **Linhas 3 e 7:** Delimitam a **cabeça** da página, local onde são realizadas algumas configurações iniciais como formatos, estilos, ícone de favoritos, etc.
- **Linha 4:** Adiciona ao documento atual o suporte a caracteres acentuados. Remover essa linha pode causar erros de renderização de algumas letras na tela.
- **Linha 5:** Indica que o conteúdo aparecerá, por padrão, ocupando todo o espaço disponível da tela e com uma escala de 1:1.
- **Linha 6:** Configura o título da página, que aparecerá como identificação da aba do navegador, ao lado do favicon.
- **Linha 8 e 10:** Delimitam o **corpo da página**, a maior porção do site, que vai aparecer na tela. É aqui onde colocaremos todo o nosso conteúdo.

HTML e CSS não é programação. Javascript é?

Quando estamos criando código HTML e CSS, podemos dizer que estamos “Desenvolvendo em HTML”, mas é errado dizer que estamos “programando em HTML”. HTML e CSS são linguagens, mas não são linguagens de programação. Essas linguagens são mais classificadas como linguagens de marcações.



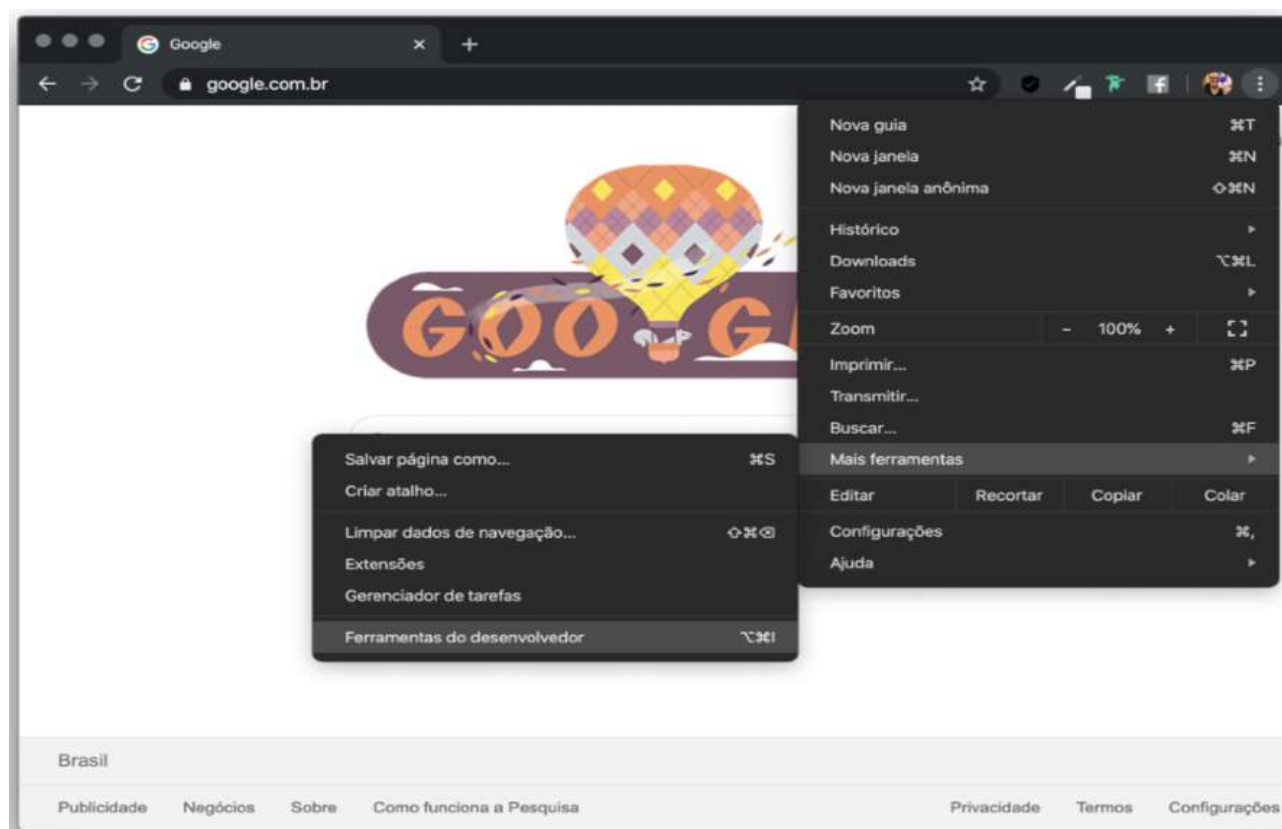
Já o JavaScript é uma linguagem de programação, considerada uma linguagem de scripts não compilados. Todos os códigos JS são enviados para o navegador do cliente e são interpretados linha-a-linha, gerando o resultado.

Sendo assim, podemos dizer “Estou programando em JS”, mas não podemos dizer “estou programando em HTML”.

Sacou? 😊

Me mostra um exemplo desse tipo de interatividade com JS?

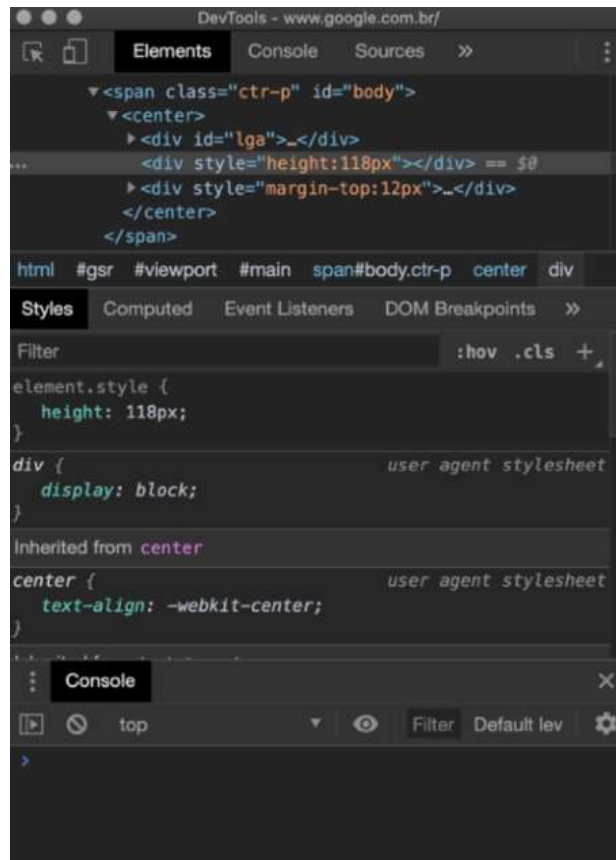
Abra aí seu Google Chrome (tem que ser ele, pois tem ferramentas que ajudam programadores como nós) e acesse o site www.google.com.br



Depois clique naqueles três pontinhos que existem lá em cima na direita (veja a imagem), escolha o menu Mais Ferramentas e depois Ferramentas do desenvolvedor.

CURSO DE VERÃO: VIAGEM AO MUNDO DA PROGRAMAÇÃO

A tela que vai abrir se chama **DevTools** e está representada aí ao lado. Ela pode aparecer colada ao seu navegador ou flutuando sobre a tela. Você muda isso clicando sobre os três pontinhos lá em cima na direita e mudando o Dock side. Mude essa opção para Undock e deixe ela sempre flutuando, vai ajudar bastante na visualização das coisas na maioria dos casos.




Essa ferramenta é muito útil para quem desenvolve sites. Como você pode perceber, ele mostra os elementos HTML, CSS e permite identificar detalhes do código.

É com o DevTools que você também vai ser capaz de detectar erros nos seus códigos que você vai começar a criar agora. Ele com certeza será um dos seus melhores amigos na caminhada por aprender a programar em JavaScript.

Mas o que mais importa no momento é a parte de baixo da tela, chamada **Console**. Ele permite que possamos interagir com componentes do site aberto no momento.

Deixe o site do Google aberto, vá até a janela do DevTools, procure o Console e digite o código a seguir:

`Window.alert('Olá Mundo!')`

Depois aperte Enter e veja a magia acontecendo. Você conseguiu fazer o site do Google falar contigo! 

Achei meio bobo 😞. Tem como melhor?

Você nunca está satisfeito(a) mesmo, não é? Pois agora você vai ser capaz de mudar o site do Google como nunca imaginou antes. Clique com o botão Ok do alerta que apareceu para fechá-lo e volte para a sua janela de console. Dessa vez, o comando vai ser um pouco maior, mas seu efeito vai ser mais legal! Confia em mim!

```
Document.body.style.backgroundColor = 'black'
```

Eu não preciso te ensinar JS agora, mas só de olhar o comando acima e você vai perceber que praticamente vai conseguir prever o que vai acontecer. Basta ter um conhecimento básico de Inglês. E se você não conseguiu entender, não se desespere! Isso vai melhorar com o tempo.

Você viu isso? O site do Google ficou preto! 🐼

Mas não sai por aí dizendo que você hackeou o site do Google! Na verdade, você só alterou a configuração do código que foi enviado para o seu computador. Qualquer outra pessoa que esteja acessando o Google da sua casa, vai ver o site normal, com o fundo branco.

Você já tem seu editor de código instalado?

Se por acaso você não possui um editor instalado, vamos recomendar o uso do **Visual Studio Code** da Microsoft no seu computador para acompanhar esse curso.



Você pode usar qualquer editor de código se a sua preferência for o Atom, Sublime, Brackets, Notepad++ ou qualquer outro, mas eu optei pelo uso do **VS Code** para preparar esse curso.

Nosso primeiro exercício JavaScript

Agora que você já criou seu primeiro exercício JavaScript, vamos ao código.

Note que a maneira mais simples de usar JS é incluindo o script em um documento HTML. Sendo assim, você já consegue identificar e classificar quase todas as linhas do documento acima. Apenas as **linhas 10, 11 e 12** são uma novidade!

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport"
6          content="width=device-width, initial-scale=1.0">
7      <title>Primeiro JS</title>
8  </head>
9  <body>
10     <h1>Primeiro JavaScript</h1>
11     <script>
12         window.alert('Olá, Mundo!')
13     </script>
14 </body>
15 </html>
```

Na recomendação atual, os scripts JS devem ser incluídos no final do corpo do documento HTML, posicionando a tag `<script></script>` logo acima do fechamento da tag `</body>`.

Tudo aquilo que estiver dentro da tag de script será considerado automaticamente JavaScript pelo seu navegador.

Nosso primeiro comando é:

`Window.alert('Olá, Mundo!')`

A palavra `window` (optional) é um objeto que indica uma referência à janela atual do navegador. A palavra `alert()` é um método capaz de disparar uma mini-janela de alerta para avisar algo ao usuário.

Dentro dos parênteses do método `alert()`, temos uma **string** entre aspas (que podem ser 'aspas simples', "aspas duplas" ou `crases` para delimitar as palavras que efetivamente aparecerão na janela de alerta).

Para executar nosso script, basta seguir o mesmo procedimento usado para abrir um arquivo HTML: vá até a pasta onde o documento está salvo, clique com o botão direito do mouse sobre o arquivo nomedoarquivo.html e escolha a opção **Abrir com Google Chrome**.

Note que assim que o arquivo abre, a mensagem de alerta já aparecerá automaticamente, antes mesmo de exibir o título <h1> que foi colocado até mesmo antes da tag <script>.

Quando você apertar o botão OK do alerta, a página vai aparecer automaticamente.

Não funcionou? Não se desespere!



Quando estamos começando a desenvolver coisas em JavaScript, é muito comum cometer pequenos erros de digitação que vão fazer nosso código simplesmente parar de funcionar.

Dependendo do deslize, pode ser que uma única linha pare de funcionar ou até mesmo o script inteiro! Mas tem um jeito de usar o **Google Chrome** para ver o que aconteceu de errado.

Olha bem a linha 11 do código que eu coloquei na página anterior. As palavras window.alert estão todas juntas, todas em minúsculo e não possuem nenhum espaço entre elas. Pois é EXATAMENTE assim que você tem que digitá-las! Qualquer pequeno deslize ou falha vai acarretar em um ERRO e seu script simplesmente não vai funcionar e a janela de alerta não vai aparecer!